02

Constructor

Transcripción

[00:00] Nuestra cuenta ahora ya está con los atributos totalmente encapsulados. En este caso aquí nos faltó darle un private a nuestro objeto cliente, pero ya sabemos que debido a nuestros métodos get y set, nuestros atributos privados no pueden ser accesibles directamente.

[00:21] Esto quiere decir que nuestro objeto es creado y no es manipulable después de ser creado, a no ser que sea a través de nuestros métodos ya definidos. Esto se le llama el estado consistente de un objeto cuando él sigue todas las reglas del negocio y en este caso pues cumple con su objetivo, que en este caso es guardar misión datos, hacer las validaciones necesarias para la manipulación de esos datos y devolverme pues esa información que yo he guardado aquí.

[00:57] Ahora, haciendo una pequeño prueba ya habíamos dicho que set agencia iba a aceptar solamente números de agencia mayores a cero, pero hay un problema con esto y vamos a reproducirlo aquí. Vengo aquí a mi clase de ejemplo y pongo el método main, y aquí le voy a decir que me cree una nueva cuenta. Cuenta = new "Ctrl + espacio". Cuenta. Perfecto. Tenemos una nueva cuenta.

[01:32] Ahora, ¿qué detalle tenemos? Sabemos que si yo le doy un cuenta.setAgencia y le pongo no sé, un -4, él no debería asignarle ese valor a mi cuenta porque yo estoy diciéndole aquí que solo me acepte agencias mayores a cero, entonces yo no debería poder ponerle ni cero ni ningún número negativo.

[02:00] Si yo le doy un system.out.print y le digo: "cuenta, dame tu agencia". Guardamos, ejecutamos, damos okay, por sí, y nos está diciendo que no se permiten valores negativos. Y sin embargo tenemos la noticia de que el valor cero está asignado a nuestra agencia. ¿Esto es por qué? Porque el valor de agencia aquí, al no ser inicializado, es creado con su valor por defecto del entero, que en este caso es igual a cero. De misma forma con números.

[02:42] Entonces aquí ya tenemos una regla del negocio que está siendo digamos vulnerada, porque no está pasando por este filtro de aquí y no tenemos forma de tener control sobre esto hasta el momento porque apenas creamos la nueva cuenta. Este objeto ya nace con ese valor por defecto, cero.

[03:04] ¿Cómo nosotros podemos resolver ese caso de uso y hacer la validación necesaria para que siga respetando el hecho de no usar negativos o cero en la agencia? Vamos a partir desde el inicio. Vamos a nuestra clase cuenta y existe un código que es ejecutado en este punto pero que nosotros no lo estamos viendo aún, y es este de acá. Es un public cuenta, entre paréntesis, abrimos y cerramos llaves.

[03:46] Y eso es todo. Este código que está aquí, cuenta, no tiene un tipo de retorno, no es un método void, no es un método int o doble o cualquier tipo de dato. Es un método que nos va a retornar nuestro objeto cuenta. Diego, ¿por qué no veíamos este código antes aquí? Bueno, en este caso, cuando solamente es este código de aquí, ya va y lo autogenera por defecto.

[04:20] Pero si nosotros hacemos algo así como System.out.print y le damos que aquí se crea una nueva cuenta, guardamos aquí y ejecutamos nuevamente y vemos claramente que el mismo texto que hemos puesto aquí, dentro del constructor, ya se está imprimiendo aquí. Y sí, el nombre de este método es constructor. ¿Por qué?

[04:53] Porque a través de este método de aquí, en el cual nosotros definimos nuestro objeto cuenta, nosotros ya podemos manipular el objeto desde su

nacimiento, desde su concepción. Esto ya nos da una idea de qué podemos hacer aquí adentro para no permitir que el número de agencia sea negativo.